CODING LOGIC - MINI PROJECT 2

Payments Management System

githubLink :- https://github.com/aomwankhede/PaymentsManagementSystem

Core Logic :-

1) There are three types of users namely :- **Admin , Financial Manager , Users** with each of the groups having different levels of access over the application.

2) Admin have access to perform all the CRUD operations on both payments and users. Financial managers can view users but can not edit them but they have complete access over the payments,and the users can also view the users and payments but can't create or edit them.

3) To login , you need to register yourself first , the validation rule is that the length of name should be greater than or equal to two and email should be valid. There is a drop down which will decide the role that is created.

4) I have made a registration form just for the first user who will register himself as admin. **The registration form doesn’t make sense here because the user is always created by admin** in this use case but the problem was who will create the first admin , that's why i have kept the registration form for simple setup and review.

5) You can see the list of users , and can filter the list based on it. For example, we want to see all the active users ,filters on the basis of types and status are present. If you are an

admin user then you can also add users by clicking on the top right button.

6) In the payments section , we can see the list of payments. The authorized individual can view , edit or delete payments by clicking on the buttons in the action column. On clicking on the view button , detailed information about that particular payment will be shown.

7) All the routes are protected and require auth so without logging in you can't see any payments list or users list , you will be redirected to the login page if you tried.

8) I have used **tailwind** for css , **vuex** for state management and **typescript** for maintaining type safety.

9) Unit tests of core components like Navbar , PaymentList , PaymentDetail , PaymentForm , UserList are written using **jest**. *For persistence, **local storage** of the browser is used.*

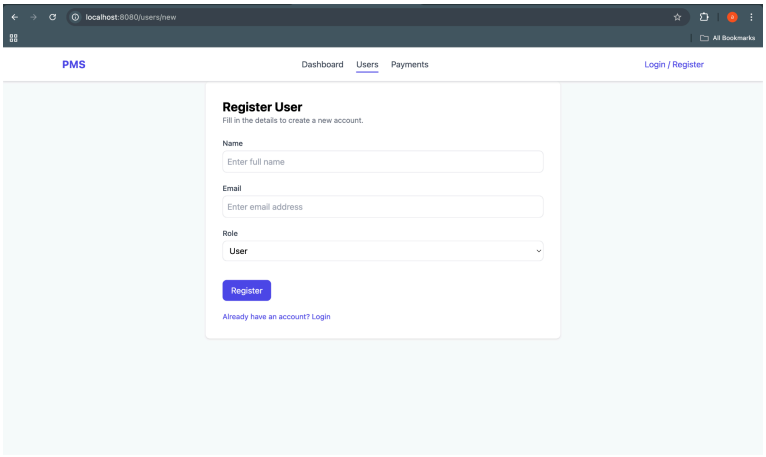
10) The user forms for registration , login , editing are using the same userForm component , they are getting reused similarly for payments.

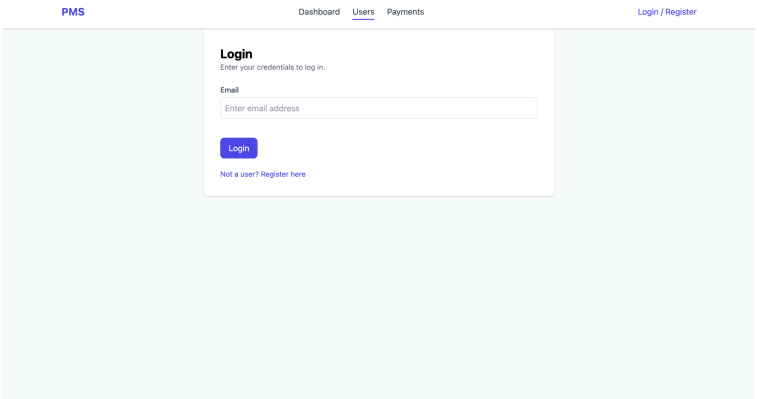
11) The design has been made sure to be **responsive** for mobile phones also.

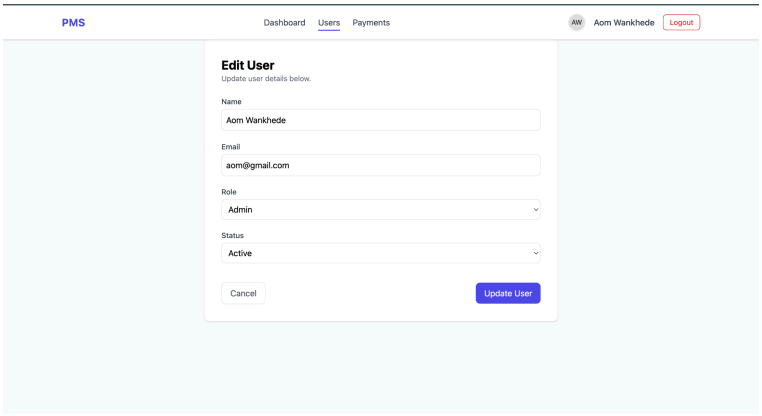
12) For setup instructions please refer to the **readme** file of the repository given.

Screenshots :-

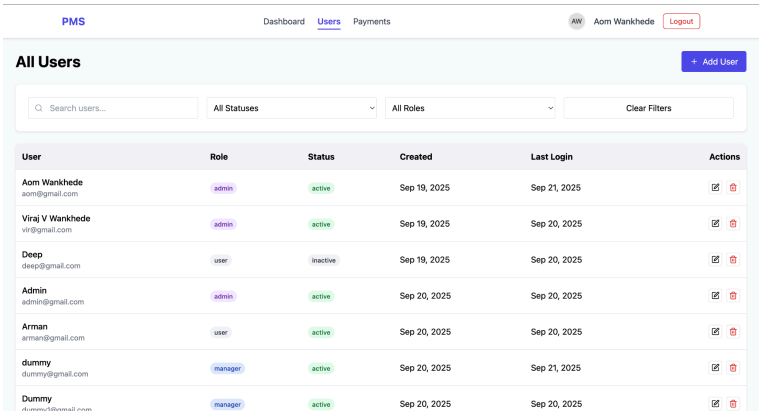
1) User Forms :-



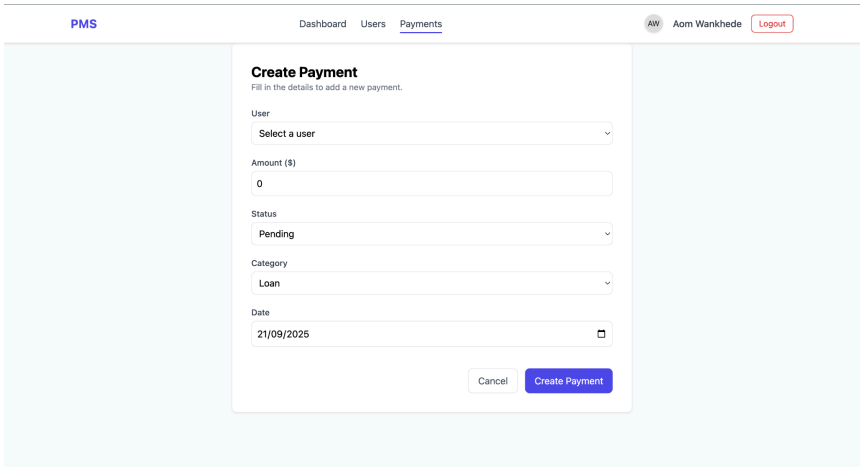


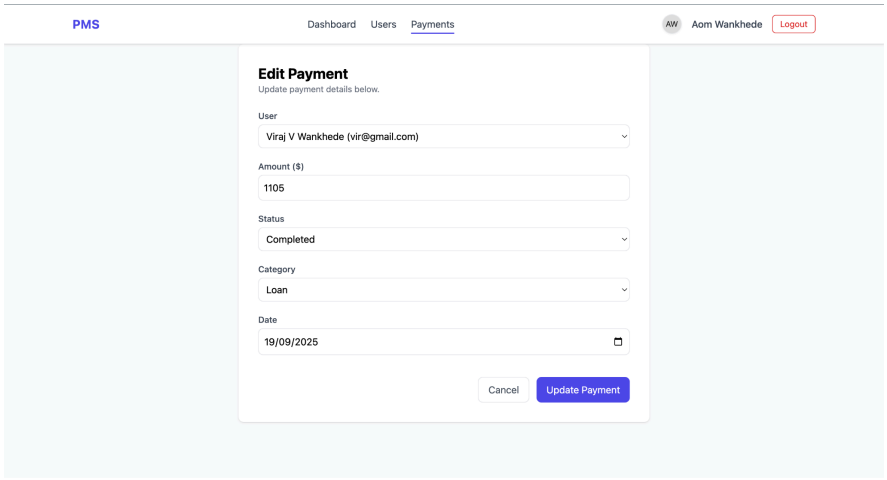


2) UserList :-

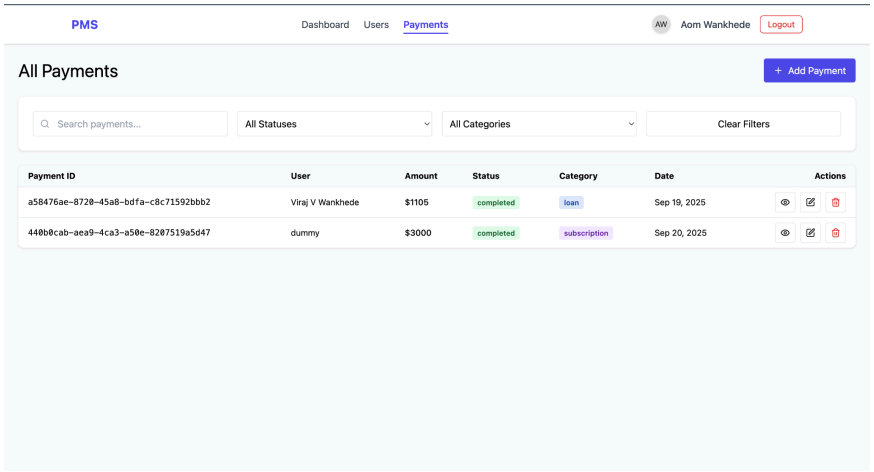


3) PaymentForm :-

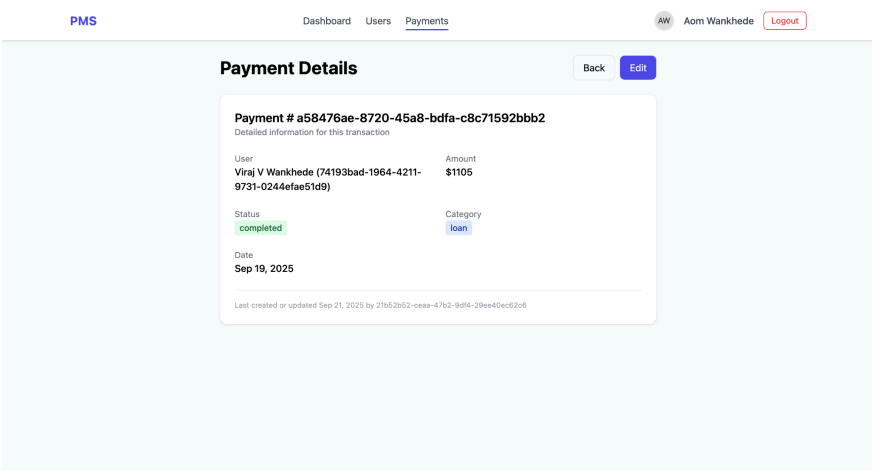




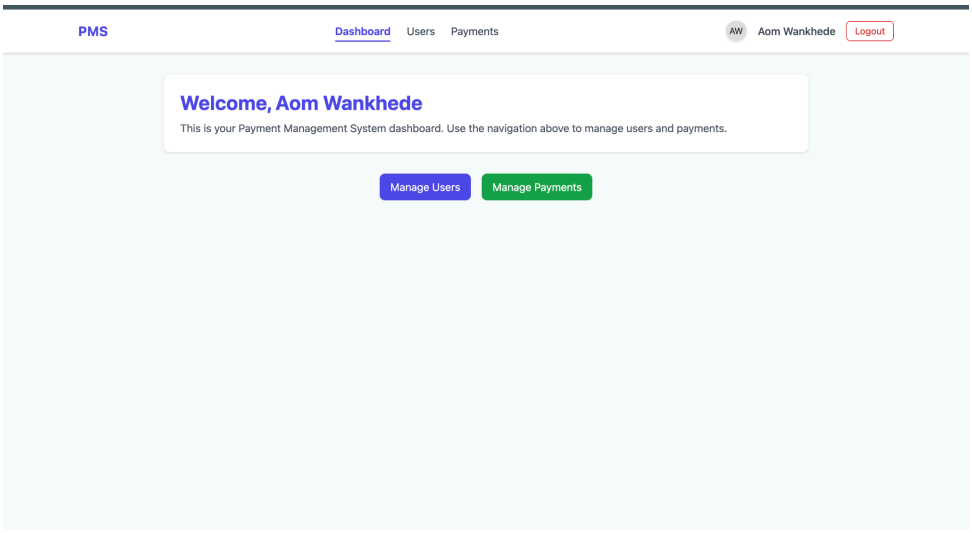
4) PaymentList :-



5) PaymentDetails :-



6) Home Page :-



Thank you !!